A Near-Optimum Channel Access Protocol Based on Incremental Collision Resolution and Distributed Transmission Queues

Rodrigo Garcés and J.J. Garcia-Luna-Aceves
Computer Engineering Department
School of Engineering
University of California
Santa Cruz, CA 95064, USA
garces@cse.ucsc.edu jj@cse.ucsc.edu

Abstract

We introduce a new stable multiple access protocol for broadcast channels shared by multiple stations, which we call the incremental collision resolution multiple access (ICRMA) protocol. ICRMA dynamically divides the channel into cycles of variable length; each cycle consists of a contention period and a queue-transmission period. The queue-transmission period is a variable-length train of packets, which are transmitted by stations that have been added to the distributed transmission queue by successfully completing a collision-resolution round in a previous contention period. During the contention period, stations with one or more packets to send compete for the right to be added to the data-transmission queue using a deterministic tree-splitting algorithm. A single round of collision resolution is allowed in each contention period. Analytical results show that collision resolution in ICRMA is much more efficient than DQRAP's. Simulation and analytical results show that ICRMA's throughput is within 5% of the throughput achieved by the ideal channel access protocol based on a distributed transmission queue and incremental collision resolution.

I. Introduction

Many of the medium access control (MAC) protocols for wireless LANs proposed to date are based on a collision avoidance dialogue between senders and receivers, e.g., [1], [2], [4], [6], [9]. A sender sends a request-to-send (RTS) to the receiver, who in turn sends a clear-to-send (CTS) if it receives the RTS free of errors; only then can the sender transmit a data packet. These protocols solve collisions by backing off and rescheduling RTS transmissions. As with CSMA protocols, this procedure yields good results if the RTS traffic is low, but is inherently unstable. As the RTS transmission rate increases, the constant RTS collisions can cause the channel to collapse, bringing the flow of data packets to a halt when no new data transmission queues can be started.

A way to stabilize the system is by increasing the retransmission delays; however, a more efficient way can be devised by using collision resolution. Several stable MAC protocols have been proposed in the past based on tree-splitting algorithms for collision resolution (e.g., [5], [7], [12]). Those protocols in which data packets are used to resolve collisions achieve throughput below 0.6 [14]. Several MAC protocols have been proposed that implement collision resolution using either control packets that are much smaller than data packets, or are based on the ability of the transmitter to abort transmission rapidly after detecting collision (e.g., [3], [8], [10]). Among those stable MAC protocols that achieve high throughput, some build a separate queue for the transmission of data packets, in addition to the stack or queue of the control packets used for collision resolution.

We introduce the incremental collision resolution multiple access (ICRMA) protocol that operates with a collision-resolution stack for control packets and a distributed queue for data packets. ICRMA does not require time slotting and mini-slots to operate like prior similar protocols did (e.g., TRAMA [10], DQRAP [15], and the announced arrival protocol [13]) and does not require base stations able to understand simultaneous transmissions (e.g., TRAMA [10]). ICRMA builds a distributed transmission queue dynamically using a deterministic tree-splitting algorithm. A station attempts to join the transmission queue during contention intervals by sending an RTS to any intended receiver, who sends a CTS if the station can join the queue. RTSs are sent according to a deterministic tree-splitting algorithm that resolves all the requests for the queue that arrive during the same contention interval. Access time to the channel is divided into rounds of transmissions for all members of the transmission queue, which we call a queue-transmission period, followed by short contention periods during which stations attempt to join the queue. The queue-transmission period is a variable-length train of packets from stations that have been added to the transmission queue by successfully completing a collision-resolution round in a previous contention period. A single step of collision resolution (i.e., a success, and idle step or a collision of control packets) is allowed in each contention period. The control packets used in each contention period are much smaller than data packets.

This work was supported in part by DARPA under Grants DAAB07-95-C-D157 and DAAH04-96-10210

Section II describes ICRMA. For simplicity, we first describe ICRMA in detail assuming a fully connected network, and then outline how to apply ICRMA to wireless LANs with hidden terminals. Section III provides the average number of steps required by a deterministic tree-splitting algorithm to resolve m collisions. We show that the number of collision-resolution steps needed in ICRMA is much smaller than the corresponding number of steps needed in DQRAP [15], which is one of the best-performing MAC protocols based on collision resolution proposed to date. We also derive new upper bounds on the number of collision-resolution steps needed in ICRMA that are independent of the number of stations and are tighter bounds than those reported by Garces and Garcia-Luna-Aceves [8]. Using these bounds, Section IV provides a lower bound on the average throughput achieved by ICRMA; analytical and simulation results show that the throughput in ICRMA approaches the channel capacity as propagation delays and size of control packets decrease with respect to the size of data packets. The analytical results are very close to the results obtained by simulation, which validates the approximations made in our analysis. Section V compares the throughput achieved in ICRMA with that achieved with an ideal channel access protocol based on transmission queues and perfect collision resolution implemented using RTS/CTS exchanges, in which only m successes are needed to resolve m requests for the transmission queue. This analysis shows that, for a given size of control packets and data packets, ICRMA's throughput is within 5% of the optimum throughput.

II. ICRMA

A. Basic Operation

With ICRMA, the channel access time is divided into cycles, with each such cycle consisting of a small, dynamically-sized contention period and a dynamically-sized queue-transmission period. Stations are assumed to monitor the state of the channel while they are not transmitting, and know the current state of the channel.

The queue-transmission period consists of the collision-free transmissions from the stations with a position in the transmission queue. Each station allocated to the transmission queue transmits a packet as soon as the packet from the previous station in the queue is received. The maximum spacing between packets is twice the propagation delay. A station wishing to send one or more packets has to acquire a position in the transmission queue before transmitting its data packets.

Stations compete to reserve a position in the transmission queue during the contention periods between queue-transmission periods. Stations follow a non-persistent CSMA strategy for the transmission of RTSs with which they request to be added to the transmission queue. The sender of an RTS to an intended destination listens to the channel for one maximum round-trip time plus the time needed for the destination to send a CTS. If the CTS is not corrupted and is received within the time limit, the station is added to the transmission queue, starting with the next queue-transmission period.

The transmission queue can be allocated a maximum size to provide delay guarantees. Once the transmission queue reaches its maximum length, stations are not allowed to request to be added to the queue, until at least one member of the queue ends transmitting its packets and leaves the queue.

Although each station transmits an RTS only when it determines that the channel is free during a predefined amount of time in the contention period, collisions of RTSs may still occur due to propagation delays. RTSs are vulnerable to collisions for time periods equal to the propagation delays between the senders of RTSs. ICRMA uses a deterministic tree-splitting algorithm to resolve collisions of RTSs. A single step of this algorithm is taken during each contention period; a step in a contention period can be an idle period, an RTS collision period, or a successful RTS/CTS exchange, all of which are much shorter than a single data

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
1998		**00-00-1998 to 00-00-1998**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A Near-Optimum Channel Access Protocol Based on Incremental Collision Resolution and Distributed Transmission Queues	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California at Santa Cruz,Department of Computer Engineering,Santa Cruz,CA,95064**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**8**	

packet.

Once the collision-resolution algorithms is started by the collision of RTSs from a set of stations, other stations who want to be added to the transmission queue must wait for all the stations who started the current round of the collision resolution algorithm to be added to the transmission queue through a successful RTS/CTS exchange; they know when this occurs by means of a stack that is maintained distributedly, which is described subsequently. We assume a non-persistent policy in which stations who are not in the transmission queue and are not involved in a collision-resolution round must backoff for a random period of time if they receive local packets to send and need to be added to the transmission queue.

Based on its knowledge of the state of the transmission queue and the current step of the collision-resolution algorithm used to add stations to the queue, a station can be in one of the following states:

- XMIT: the station is part of the transmission queue.
- RTS: the station is trying to acquire a position in the transmission queue and is participating in the collision-resolution algorithm.
- BACKOFF: the station is waiting for a backoff time to complete as a result of receiving local packets to send outside an access period.
- REMOTE: the station is not part of the transmission queue and knows that a set of collisions of requests for the queue are still being resolved.
- PASSIVE: the station is not part of the transmission queue, has no local packets to send, and knows that no prior requests for the queue are being resolved.

B. Information Maintained and Exchanged

Each station is assigned a unique identifier, knows the maximum number of stations allowed in the network and the maximum propagation delay, and maintains a stack and two variables ($LowID$ and $HiID$).

$LowID$ is initially set to 1 and denotes the lowest ID number that is allowed to send an RTS. $HiID$ is the highest ID number that is allowed to send an RTS and is initially set to the largest number of stations allowed in the network. $LowID$ and $HiID$ constitute the allowed ID-number interval that can send RTSs. If the ID of the station is not within this interval or the station has already been assigned to the data transmission queue, it cannot send its RTS. The stack is simply a storage mechanism for ID-intervals that are waiting for permission to send an RTS.

Each station also maintains the state of the transmission queue, i.e., it knows the members in the transmission queue, their position in the queue, and the beginning of each queue-transmission period. RTSs and CTSs specify the IDs of the sender and the intended receiver, as well as the allowed ID-number interval known to the sending station. Including the allowed ID interval in control packets allows listening stations to be updated on the state of the collision resolution. A data packet contains: (a) the user data and destination of the packet; (b) the allowed ID-number interval known to the sending station; (c) the state of the transmission queue, such as the number of stations in the queue and the position of the sending station in the queue; and (d) the state of the transmission for the sending station, such as the length of the current packet, whether the station is leaving the transmission queue, and the length of the *next* packet if variable length packets are allowed. This information updates listening stations quickly on the state of the channel.

C. Adding Members to the Transmission Queue

Stations can request to be added to the transmission queue only during contention periods. If the maximum length of the transmission-queue is reached, the contention steps are skipped, until space is made available by one or multiple stations leaving the queue. Stations are allowed to send RTSs to be added to the transmission queue only during the first few seconds of a contention period. This time period is called the *access period* and is used in order to avoid collisions of RTSs with the beginning of the transmission queue. For the purposes of analysis, we assume subsequently that the access period lasts τ seconds, but it is a configurable parameter. A contention period consists of a single step of a deterministic tree-splitting algorithm based on the allowed ID interval, which we describe subsequently.

If a PASSIVE station receives local packets to send during an access period (the first τ seconds of a contention period), it first listens to the channel. If the channel is clear (i.e., no carrier is detected), the station enters the RTS state by sending an RTS. The sender then waits and listens to the channel for one maximum round-trip time plus the time needed for the destination to send a CTS. When the originator receives the CTS from the destination, it is added to the data transmission queue and enters the XMIT state; the station can start sending collision-free packets in the following queue-transmission period.

On the other hand, if a PASSIVE station obtains local packets to send outside an access period, or receives the packets within an access period but detects carrier, it enters the BACKOFF state. The station then computes a random backoff time and attempts to enter the data transmission queue after that time.

If the sender of an RTS does not receive the corresponding CTS within the allocated time, this station and all other stations in the network assume that a collision has occurred. The collision-resolution algorithm is started with the first round of RTS collisions. As soon as a collision is detected, every station divides the ID-interval $(LowID, HiID)$ into two ID-intervals. The first ID-interval, which we will call the backoff interval, is $(LowID, \lceil \frac{HiID+LowID}{2} \rceil - 1)$, while the second ID-interval, the allowed ID interval, is $(\lceil \frac{HiID+LowID}{2} \rceil, HiID)$. Each station in the system updates its stack by executing a PUSH-stack command, where the key being pushed is the backoff interval. After this is done, the station updates $LowID$ and $HiID$ with the values from the allowed ID interval and the queue-transmission period follows. This procedure is repeated each time a collision is detected.

Hence, a station knows if collisions are currently being resolved when its local stack is not empty and the allowed ID interval does not equal the entire ID interval. A PASSIVE station that detects a collision of RTSs goes to the REMOTE state. A station in REMOTE state goes to PASSIVE state as soon as it detects that its local stack is empty and the allowed ID interval is empty. If a station is in REMOTE state and obtains one or more packets to send, it goes immediately to the BACKOFF state.

Only those stations that are in the RTS state participate in the current round of the collision-resolution algorithm to be added to the transmission queue. All stations in the allowed ID-interval that are in the RTS state transmit an RTS in the next contention period. Stations outside of the transmission queue that did not start the current round of collision resolution are forced into REMOTE or BACKOFF state.

Each step of the collision-resolution algorithm, and consequently each contention period, can be one of the following three cases:

- Case 1–*Idle*: There is no station in the RTS state whose ID is within the allowed ID interval. Therefore, the channel must remain idle one collision-resolution step and no new member is assigned to the transmission queue. An idle step of collision resolution lasts a maximum round-trip time; after that time, the stack and the variables $LowID$ and $HiID$ are updated; each station executes a POP-stack command. This new ID-interval now becomes the new $HiID$ and $LowID$.
- Case 2–*Success*: There is a single station with an RTS to send whose ID lies within the allowed ID interval. In this case, a single station is able to complete an RTS/CTS handshake successfully, enters the XMIT state, and is added to the transmission queue. The duration of a successful collision-resolution step lasts two RTS lengths and two channel propagation delays; after that time, the stack and the variables $LowID$ and $HiID$ are updated; each station executes a POP-stack command. This new ID-interval now becomes the new $HiID$ and $LowID$.
- Case 3–*Collision*: There are two or more stations with RTSs to send whose IDs are within the allowed ID interval; therefore, each such station sends an RTS creating a collision. Because stations use carrier sensing, a collision period can last no more than an RTS plus a maximum round-trip time. The stations in the allowed ID-interval are again split into two new ID-intervals; the stack and the variables for each station are updated (PUSH-stack command).

The tree-splitting algorithm executes a collision resolution (idle, success, or collision), until no station remains in the RTS state and all requests for the queue have been resolved. Again, all stations know when the tree-splitting algorithm terminates its current round, i.e., when the backoff stack is empty and there are no values in the allowed ID interval. Once termination is detected, the allowed ID interval is updated to include all the ID range.

To ensure fairness within the splitting algorithm, the position of the stations in the tree (which is equivalent to changing the ID number) can vary after each tree-contention period. For example, the ID numbers of the stations can rotate cyclically. Each station increases its ID number by one and the last station takes the ID number of the previous first station.

D. Deleting Members from the Transmission Queue

Deletions from the transmission queue occurs when a station in the XMIT state ends transmitting a packet train and gets out of the queue to remain quiet. In this case, a data packet notifies the stations when the sending station is sending its last packet before leaving the queue.

Deletions from the transmission queue can also occur due to failures. Failures can be handled by listening to the channel. A maximum interval allowed between two consecutive packets in the queue can be defined based on the maximum propagation delay and processing delays. Because stations are at most τ seconds apart from one another, two consecutive packets in the queue cannot be separated by more than 2τ seconds plus processing delays (including the turn around times of the radios), which we call *maximum queue transmission gap* and assume it lasts ϕ seconds.

A station in the queue must transmit when its turn comes so that its transmission reaches other stations within ϕ seconds of the previous transmission. A "silence turn" could occur in the transmission queue if a station is not ready to transmit a data packet when its turn comes but is not ready to leave the queue. Rather than allowing silence turns to take place, ICRMA requires a station to transmit either a data packet or a short control packet when its turn comes in the queue. The control packet carries no user data and is simply used to convey an update of the state of the queue and the collision resolution process.

E. Example

The following example illustrates the way in which stations are added to the transmission queue. If n is the total number of stations in the system, the binary tree used for collision resolution has $2n + 1$ nodes. We label the root of the tree as n_r and its right and left child as n_1 and n_0, respectively. For each of the other nodes, the labels are composed of the parent label, plus a 0 if it is a left child or a 1 if it is a right child. Consider the case of a network with four stations and assume that, at time t_0, station n_{11} is the only station assigned to the queue-transmission period with five data packets to transmit and the last cycle has ended and we are at the beginning of a new cycle, that is, in the contention period.

During the access period, packets arrive at stations n_{01} and n_{00}. At time t_0, a collision of RTSs occurs with station n_{00} and n_{01} each sending an RTS in the same contention period, while station n_{10} and station n_{11} do not request to be added to the data transmission queue (Step 2 in Fig. 1). Station n_{11} does not need to send a request, because it is already assigned to the data transmission queue and all other stations know that it still has five data packets to transmit. Let station n_{01} have three data packets to send while station n_{00} has only one data packet to transmit. At time t_0 the first collision of RTSs occurs, all stations in the system notice the beginning of the resolution algorithm, as well as the beginning of the contention period. All stations update their stacks and their $LowID$ as well as their $HiID$ values. After at most an RTS plus a maximum round-trip time has elapsed, the first queue-transmission period begins with station n_{11} transmitting a packet. The duration is the size of the length of the packet plus the channel delay. In the next contention period stations n_{00} and n_{01} backoff and wait until the collisions in the allowed-ID interval are resolved (Step 3 in Fig. 1). They both are excluded from sending RTSs. Stations n_{10} and n_{11} are allowed to request the channel; an idle contention period occurs because stations n_{11} and n_{10} do not wish to be added to the data transmission queue. After 2τ seconds, all stations notice that the channel is idle, which means that there were no collisions. All the stations in the system must update their intervals and the stack. They execute a POP-stack command and the new allowed interval is $(00, 01)$; therefore, after n_{11} transmits another packet, the next allowed ID interval can proceed to solve its RTS collisions. Both stations n_{00} and n_{01} transmit an RTS control packet and a collision occurs again (Step 4 in Fig. 1). Because a collision occurred, after at most an RTS plus a maximum round-trip time has elapsed the allowed ID interval is split in two. Once again the step in the contention is followed by a queue-transmission period. Station n_{11} is still in the group and has two more packets to send. In the next contention step (Step 5 in Fig. 1), station n_{01} is within the allowed interval while the n_{00} station must wait; its interval is the top of the stack. Since the next interval has only one station sending an RTS, station n_{01} is assigned to the data transmission queue after an RTS, a CTS plus a maximum round-trip time. During the successful RTS/CTS exchange, all other stations know that station n_{01} has three packets to send. The contention step ends and station n_{01} begins the transmission of its data packet immediately after it has read the packet transmitted by the preceding group member, which is station n_{11}. After two data packets plus a maximum round-trip time has elapsed, station n_{11} has one data packet left while station n_{01} has two packets. The next contention step is also a successful RTS/CTS exchange (Step 6 in Fig. 1); after an RTS, a CTS plus a maximum round-trip time, station n_{00} is added to the group and the allowed interval as well as the stack are empty. The termination of the tree-splitting algorithm is reached when the backoff stack as well as the allowed ID interval is empty. The data transmission queue contains a packet from stations n_{00}, n_{01} and n_{11}. After the last group-transmission period stations n_{00} and n_{11} are done sending packets and free the space within the group. If new packets arrived at one of these stations they will have to request a space in the group. Let us assume that that after termination of the tree-resolution period no new data packets arrive during the access period, therefore, the channel remains idle for two channel delays. During the next queue-transmission period, station n_{01} sends the last packet. Station n_{01} is removed and the data transmission queue is empty. The channel remains idle until a new packet arrives. The allowed ID interval is (n_{00},n_{11}) and the backup stack is empty.

F. Handling Hidden Terminals

Handling hidden terminals in ICRMA is more difficult than in other MAC protocols based on collision avoidance because all stations whose transmissions can impact one another must agree on the same state of the channel and the transmission queue, rather than just agreeing on allowing a given sender to transmit a packet without interference.

Fullmer and Garcia-Luna-Aceves [6] have demonstrated sufficient conditions for MAC protocols based on RTS/CTS exchanges to eliminate unwanted collisions due to hidden terminals. For the case of MAC protocols that use carrier sensing, a CTS must be longer than the length of an RTS, plus a maximum round-trip time and processing delays (including radio turn-around times). The net effect of the CTS length suggested in [6] is that of a single-channel, receiver-based busy tone that forces hidden sources to back off after they are forced to listen to at least part of the CTS sent by a receiver that has accepted the RTS from a given sender. Note, however, that a CTS can be only partially overheard by a station after it sends a failed RTS, and therefore it cannot be used to convey queue and channel state information to senders hidden from one another.

Consider the case of a wireless local area network (WLAN) in which stations are trying to communicate with one another over a single hop, or through a base station. In this case, each sender is in line of sight of the intended receiver (another station or the base station) and all stations should agree on the same state of a single transmission queue and the same state of the channel. We call this case a simple WLAN.

For ICRMA to work correctly in a simple WLAN, the base station is the one that should send the CTSs in oder for all hidden senders to hear that a transmission request has been granted. Furthermore, to notify the correct state of the queue, the base station must also send a second control packet, which we call the *channel and queue state* packet (CQS) immediately after it sends a CTS. The CQS has no length restrictions relative to an RTS and contains all the information on the state of the queue and channel as perceived by the receiver. The minimum channel information needed in a CQS consists of the ID of the sender of the successful RTS.

III. Average Number of Collision Resolution Steps

In this section, we present upper bounds for the average number of steps needed for a collision resolution algorithm based on deterministic tree-splitting algorithm to resolve m collisions in a network of n stations, each having a unique ID. We assume that: (a) every station can listen to every other station; (b) the channel introduces no errors, so packet collisions are the only source of errors; (c) stations detect such collisions perfectly; (d) two or more transmissions that overlap in time in the channel must all be re-transmitted; (e) a packet propagates to all stations in exactly τ seconds [11]; and (f) no failures occur.

Because each station is assigned one or zero RTS at any given time, stations in the network are assigned an "RTS" or an "idle," depending on whether or not they have an RTS to send.

As we have described, there are only three types of collision-resolution steps: idle, success, or collision. $\overline{\mathcal{Z}}(n,m)$ denotes the average number of idle steps, $\overline{\mathcal{C}}(n,m)$ denotes the average number of collision steps, and $\overline{\mathcal{S}}(n,m) = m$ is the number of successful steps of the collision-resolution algorithm needed to resolve m collisions in a network of n stations.

Garces and Garcia-Luna-Aceves [8] obtained the following recursive equations for these averages, which are obtained by considering all possible collision-resolution trees resulting from allocating m RTSs over n stations:

$$
\begin{aligned}
\overline{\mathcal{Z}}(n,m) &= \sum_{i=\mu}^{\nu} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m-i) + \mathcal{Z}(\beta, i)\right] \\
\overline{\mathcal{C}}(n,m) &= \sum_{i=\mu}^{\nu} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\overline{\mathcal{C}}(\alpha, m-i) + \overline{\mathcal{C}}(\beta, i) + 1\right] \\
\alpha &= \lceil n/2 \rceil;\ \beta = n - \alpha = n - \lceil n/2 \rceil \\
\mu &= \begin{cases} 0 & \text{if } m \leq \alpha \\ m - \alpha & \text{if } m > \alpha \end{cases} \\
\nu &= \begin{cases} m & \text{if } m \leq \beta \\ \beta & \text{if } m > \beta \end{cases}
\end{aligned} \tag{1}
$$

We can now compare the average number of collision-resolution steps required in ICRMA with the steps required in DQRAP [15], which is a competitive protocol based on collision resolution. For this comparison, we assume in ICRMA a cost of one step per collision resolution step regardless if the step was a collision, a success or an idle step. The total number of stations for both protocols is $n = 20$; therefore, the maximum number of stations that can collide is $m = n = 20$. The average number of steps needed in DQRAP was calculated using the equation for collision resolution interval (CRI) length $L(m, s)$ provided in [15] and shown in

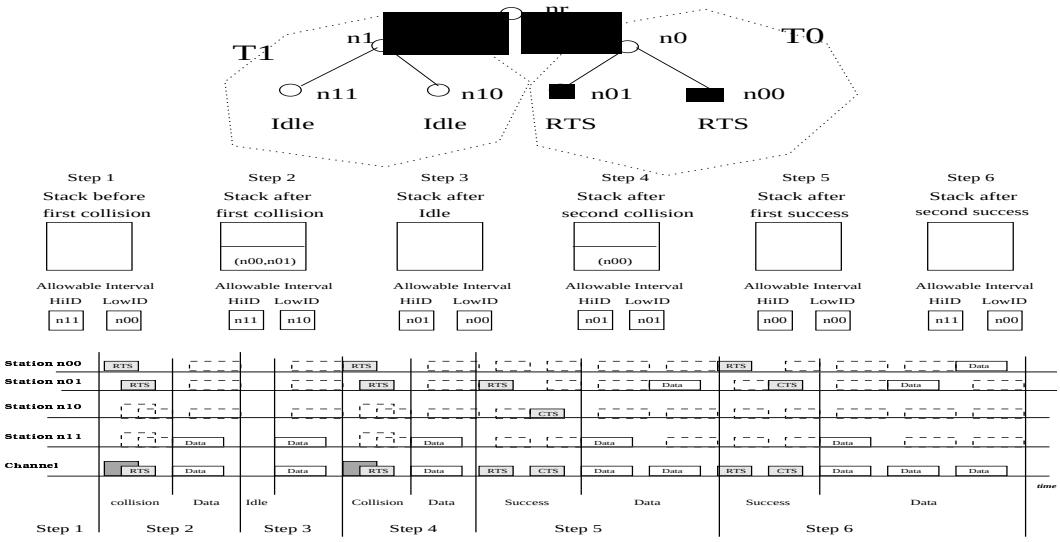

Fig. 1. Tree structure for an example with $n = 4$ stations and $m = 2$ stations, station n_{01} and n_{00}, requesting to be admitted into the data transmission queue. Station n_{11} is already in the queue.

Eq. (2) which is a function of the number of station colliding in the previous CRI (m) and the number of mini-slots per CRI called s:

$$L(m,s) = \frac{s^{m-1}}{s^{m-1}-1}\left[1 + \sum_{k=2}^{m-1}\binom{m}{k}\frac{(s-1)^{m-k}}{s^m}L(k,s)\right] \qquad (2)$$

Because the number of mini-slots per CRI is s, the total number of steps needed in DQRAP to solve m collisions is equal to the number of mini-slots per CRI times $L(m,s)$. Fig. 2 shows the total number of steps needed to resolve m collisions for both DQRAP and ICRMA, when $n = 20$. DQRAP was calculated using different number of mini-slots per CRI. The figure illustrates that if the same mechanism is used in ICRMA and DQRAP to detect collisions, the collision resolution used in ICRMA renders a higher throughput than DQRAP does.

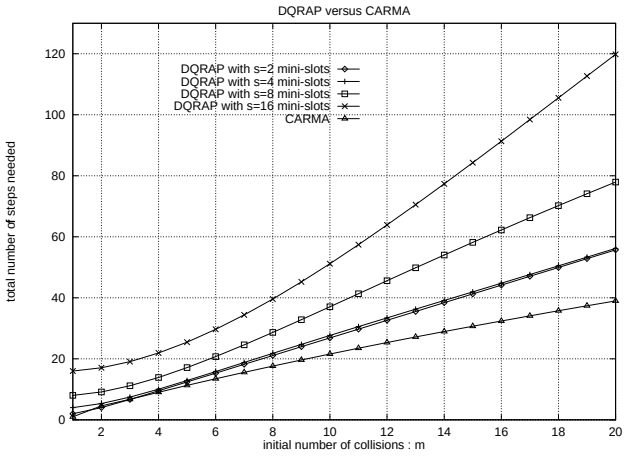

Fig. 2. Total number of collision resolution steps for DQRAP for different mini-slots, s=2,4,8,16 and the collision resolution algorithm used in ICRMA.

The following two theorems provide upper bounds on the average number of idle and collision steps needed, which are independent of the number of stations in the network. The Appendix contains the proof of these bounds.

Theorem 1: For all $n \geq m > 1$, where n is the total number of stations in the network and m is the number of stations competing for a space in the transmission queue, the average number of idle steps is bounded by $\overline{\mathcal{Z}}(n,m) \leq 0.443m$.

Theorem 2: For all $n \geq m > 1$, where n is the total number of stations in the network and m is the number of stations competing for a space in the transmission queue, the average number of collision steps is bounded by $\overline{\mathcal{C}}(n,m) \leq 1.443m - 1$.

It follows from these two theorems and the fact that m successes are needed to resolve m requests for the transmission queue that the total number of collision-resolution steps needed to resolve m RTS collisions is smaller or equal to $2.886m - 1$ steps. This clearly indicates that the type of collision resolution used in ICRMA is very efficient. Furthermore, these bounds are independent of the number of stations in the network, which we use to provide an approximate throughput analysis.

IV. Throughput Analysis

The analysis in this section uses the same traffic model used by Kleinrock and Tobagi [11] to analyze CSMA protocols. It is assumed that there is large number of stations, each forming a Poisson source of RTSs (both new and retransmitted). The aggregate mean generation rate of RTSs by all stations is λ RTSs per unit time. Each station has at most one RTS to transmit at any given time. With this model, the average number of RTS arrivals in a time interval τ is $\lambda\tau$, i.e., $m = \lambda\tau$. All data packets have a duration of δ seconds, and the time to transmit a control packet is γ seconds. Both γ and δ are multiples of τ. The number of packets in a message is a random variable, and the probability that a message will complete its transmission (in a given cycle) is given by $q = \frac{1}{N}$ where N is the average number of packets in a message. We also assume that the time to transition between transmit and receive states is negligible.

The probability P_z that a contention period is idle is equal to the probability that no packets arrive during the access period τ, which equals $P_z = e^{-\lambda\tau}$.

For a station to be added to the data transmission queue, the RTS/CTS exchange must be successful, which implies that the RTS must be the only one in the channel during its access period and the data transmission queue must not be full. If there are already h group members, no new member can be added and the contention would be skipped (no station sends an RTS), until a space in the data transmission queue is available. Therefore, the probability P_r that a station sends an RTS successfully equals the probability that only one RTS packet arrives during the τ seconds of the access period, that is, $P_r = \lambda\tau e^{-\lambda\tau}$.

The probability P_c that a contention period results in an RTS collision is the same as the probability that more than one message arrives during an access period. This value can be expressed as $P_c = 1 - P_r - P_z = 1 - \lambda\tau e^{-\lambda\tau} - e^{-\lambda\tau}$. Making use of the upper bounds in Theorems 1 and 2, the probability that an RTS/CTS exchange is successful after the tree-splitting algorithm has been started can be approximated by $P_a \approx \frac{m}{2.886m-1} = \frac{\lambda\tau}{2.886\lambda\tau-1} \approx \frac{1}{3}$, where $m \geq 2$. It is clear that P_a should be a function of both the number of stations that start a round of the tree-splitting algorithm, and the number of collision-resolution steps that have occured up to the successful step. Using an average over all steps in a round of collision-resolution is an approximation we use to simplify our analysis.

The states of the queue-transmission period can be represented by a Markov chain. The states in the Markov chain represent the probability Π_k that k members are in the data transmission queue. A transition in the chain occurs with each collision-resolution step. The number of members in the data transmission queue

can only increase by one, but can decrease by up to k. This is because at most one new member can be added to the data transmission queue per collision-resolution step, but any number of members can be released from the data transmission queue per step. Fig. 3 shows an example for a network that allows at most $h = 4$ stations to be members of the group-transmission period. Transitions from one state to the other have been adequately labeled, where $W = P_r + P_c P_a$ and $V = P_z + P_c(1 - P_a)$.

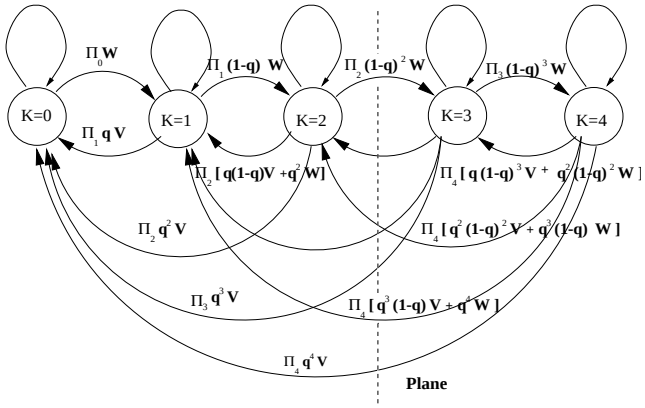

Fig. 3. Markov Chain defining the transitions from one state to the others. The given example is for a network that allows, up to four members in the data transmission queue. Only a subset of the transition probabilities are shown

We can generalize our example and define h as the maximum number of members in the data transmission queue allowed by the network. If we draw a line between any two consecutive states of the Markov chain, then the flow of the transitions going in one direction has to be equal to the flow in the other direction. Therefore, for any cut within the Markov chain excluding the first and the last state, i.e., for $1 \leq k \leq h-2$,

$$\begin{aligned}\Pi_k W(1-q)^k &= W \sum_{i=1}^{h-k-1} \Pi_{k+i} \sum_{j=i+1}^{k+i} q^j (1-q)^{k+i-j} + \\ & V \sum_{i=1}^{h-k-1} \Pi_{k+i} \sum_{j=i}^{k+i} q^j (1-q)^{k+i-j} + \\ & W\Pi_h \sum_{j=h-k+1}^{h} q^j (1-q)^{h-j} + V\Pi_h \sum_{j=h-k}^{h} q^j (1-q)^{h-j}\end{aligned} \tag{3}$$

If we divide both sides by $W(1-q)^k$, the result is

$$\begin{aligned}\Pi_k &= \sum_{i=1}^{h-k-1} \Pi_{k+i} \left(\sum_{j=i+1}^{k+i} \frac{q^j}{(1-q)^{j-i}} + \frac{V}{W} \sum_{j=i}^{k+i} \frac{q^j}{(1-q)^{j-i}} \right) + \\ & \Pi_h \left(\sum_{j=h-k+1}^{h} \frac{q^j}{(1-q)^{j+k-h}} + \frac{V}{W} \sum_{j=h-k}^{h} \frac{q^j}{(1-q)^{j+k-h}} \right)\end{aligned} \tag{4}$$

For the cut between $k = 0$ and $k = 1$ the flow equation is

$$\Pi_0 = \frac{V}{W} \left(\sum_{j=1}^{h} \Pi_j q^j \right) \tag{5}$$

For the cut between $k = h-1$ and $k = h$ the flow equation changes since the network cannot hold more than h members in the data transmission queue. There is no state to which to increase after Π_h has been reached, in fact

$$\Pi_{h-1} = \Pi_h \left(\frac{V}{W} \left(\sum_{j=1}^{h} \frac{q^j}{(1-q)^{j-1}} + \sum_{j=2}^{h} \frac{q^j}{(1-q)^{j-1}} \right) \right) \tag{6}$$

Eqs. (4), (5) and (6), together with the fact that the sum of all probabilities $\sum_{i=0}^{h} \Pi_i = 1$, form a system with $h+1$ equations with the same number of unknown variables. Starting with Π_{h-1} we can make each of the equations be a function of Π_h. To solve for Π_h, we can substitute each of these terms in the equation $\sum_{i=0}^{h} \Pi_i = 1$. Using Π_h we can get a value for each of the remaining probabilities. Π_0 is the probability that there are no members in the data transmission queue, while Π_h is the probability that the data transmission queue has reached full capacity. Π_h is important since it represents the fraction of skipped contention periods. Transitions departing from and returning to the same state are omitted in the flow equations, since what comes into the state is the same as what goes out of the state. Thus, the average number of group members is $\rho = \sum_{i=1}^{h} i\Pi_i$.

The average channel throughput is equal to

$$S = \frac{\overline{U}}{\overline{B_c} + \overline{B_g} + \overline{I}} \tag{7}$$

where $\overline{U}$ is the average utilization time of the channel, during which the channel is being used to transmit data packets; $\overline{B_c}$ is the expected duration of the contention period; $\overline{B_g}$ is the expected duration of the queue-transmission period; and $\overline{I}$ is the average idle period, i.e., the average interval between two consecutive busy periods.

Because data packets are send free of collisions, $\overline{U}$ equals on the average number of members in the data transmission queue (ρ) times δ, which is the time spent sending a data packet, that is,

$$\overline{U} = \rho\delta \tag{8}$$

A contention period is skipped if the data transmission queue is full; otherwise, if there is room in the queue, which is the case with probability $1 - \Pi_h$, a contention period can be idle, successful, or have RTS collisions. The duration of an idle contention period is $T_i = 2\tau$, the duration of a successfully RTS/CTS exchange period is $T_s = 2\gamma + 2\tau$, and the duration collision period is $T_f \leq \gamma + 2\tau$. Therefore, the duration of the contention period is at most $2\gamma + 2\tau$ when it is not idle. Accordingly, the average duration of a contention busy period can be bounded by

$$\overline{B_c} \leq [2\tau P_z + T_s(P_r + P_c)](1 - \Pi_h) \tag{9}$$

The average queue-transmission period is equal to the average number of group members (ρ) times the transmission period for one packet ($\delta + \tau$); therefore,

$$\overline{B_g} = \rho \cdot (\delta + \tau) \tag{10}$$

If the data transmission queue is empty, the length of the idle period is given by the next RTS arrival into the channel plus a waiting period of 2τ. In contrast, when the data transmission queue is not empty, the length of the idle period is limited by the start of the next transmission period, which is 2τ seconds. Accordingly, we obtain

$$\overline{I} = \Pi_0 \left(\frac{1}{\lambda} + 2\tau \right) + (1 - \Pi_0) 2\tau = \frac{1}{\lambda}\Pi_0 + 2\tau \tag{11}$$

Substituting Eqs. (8), (9), (10), and (11) into Eq. (7) the average throughput can be bounded as:

$$S \geq \frac{\rho\delta}{2\gamma(1-\Pi_h)(1-e^{-\lambda\tau}) + \tau(4 - 2\Pi_h) + \rho(\delta + \tau) + \frac{\Pi_0}{\lambda}} \tag{12}$$

The performance comparison is done for both low speed network (9600 bps) and high speed network (1 Mbps) with small data packets of 53 bytes (as in ATM cells) and longer data packets of 400 bytes. We assume the spacing between stations to be the same and define the diameter of the network to be 16.090 Km., which is 10 miles. Assuming these parameters, the propagation delay of the channel is 54μs. To accommodate the use of IP addresses for destination and source, the minimum size of RTSs and CTSs is 20 bytes. We normalize the throughput result by setting $\tau = 1$ and defining the following variables

$$
\begin{aligned}
a &= \frac{\delta}{\tau}(\text{normalized data packets}) \\
b &= \frac{\gamma}{\tau}(\text{normalized control packets}) \\
G &= \lambda \cdot \tau(\text{normalized offered load})
\end{aligned} \tag{13}
$$

If we substitute the new normalized variables from Eq. (13) into the throughput Eqs. (12), we obtain

$$
S \geq \frac{a\,\rho}{2b(1-\Pi_h)(1-e^{-G})+4-2\Pi_h+(a+1)\rho+\frac{\Pi_0}{G}} \tag{14}
$$

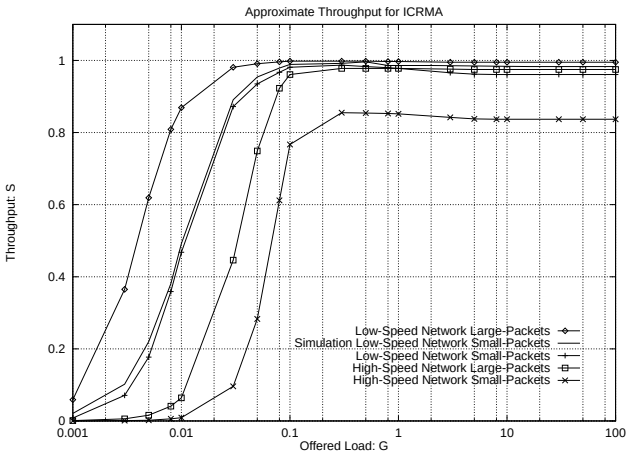

Fig. 4. Approximate throughput comparison for ICRMA. The maximum number of group members is $h = 32$ and the average number of packets in a message is $N = 10$.

Fig. 4 shows the throughput versus offered load given by Eq. (14). We consider a low-speed network at 9600 bps, as well as a high-speed network at 1 Mbps using small data packets ($\delta = 53$ bytes) and large data packets ($\delta = 400$ bytes). The throughput is for each of the four combinations. As the average number of members in the data transmission queue increases due to increases in the offered load, the throughput of ICRMA reaches a constant throughput value that, approaches the channel capacity as the relative overhead due to propagation delays and control packets approaches 0. By taking the limit as $\lambda \to \infty$, we obtain

$$
\lim_{\lambda \to \infty} S \geq \frac{h\delta}{2\tau + h(\delta+\tau)}
$$

which clearly shows that ICRMA's overhead is very small at high load. To verify that the value of S approximated using the upper bounds on average steps for collision resolution times provides a good lower bound for any traffic load, we simulated ICRMA using 128 stations that generate RTSs according to a Poisson probability distribution function. The maximum number of members allowed in the data transmission queue h was set to 32 and the average number of packets in a message was set to $N = 10$. The simulations were done ten times for each given $m = \tau\lambda$ value to insure convergence. The results of the simulation are shown in Figure 4 and indicate that our analysis provides a very good lower of the average throughput.

V. Comparison with an Optimum Transmission Queue Protocol

An optimum MAC protocol based on a distributed transmission queue and RTS/CTS handshakes for additions to the transmission queue would require exactly m successful RTS/CTS handshakes to resolve m requests for additions to the transmission queue. In this section, we obtain the average throughput for such an optimum MAC protocol using the model and results derived in the previous section and show that ICRMA is indeed very close to the best possible performance.

In terms of our analysis in the previous section, the probability P_a that an RTS/CTS exchange is successful in the contention period when space is available in the transmission queue is always 1. Therefore, the parameters W and V for the Markov chain (see Fig. 3) become $W = P_r + P_c$ and $V = P_z$. All other parameters in the Markov chain remain the same. If there are multiple stations with RTSs at the beginning of the round of collision resolution, it is assumed that the protocol organizes the transmission of RTSs in a way that success is ensured at each step. Accordingly, the collision-resolution steps for the optimum MAC protocol consist of successful RTS/CTS exchanges only and the average contention busy period in Eq. (9) can be rewritten as:

$$
\overline{B_c} = [2\tau P_z + T_s(P_r + P_c)](1-\Pi_h) \tag{15}
$$

The rest of the variables in Eq. (7) remain unchanged. Substituting Eqs. (8), (15), (10), and (11) into Eq. (7) the average normalized throughput for the perfect protocol can be written as:

$$
S_p = \frac{a\,\rho}{2b(1-\Pi_h)(1-e^{-G})+4-2\Pi_h+(a+1)\rho+\frac{\Pi_0}{G}} \tag{16}
$$

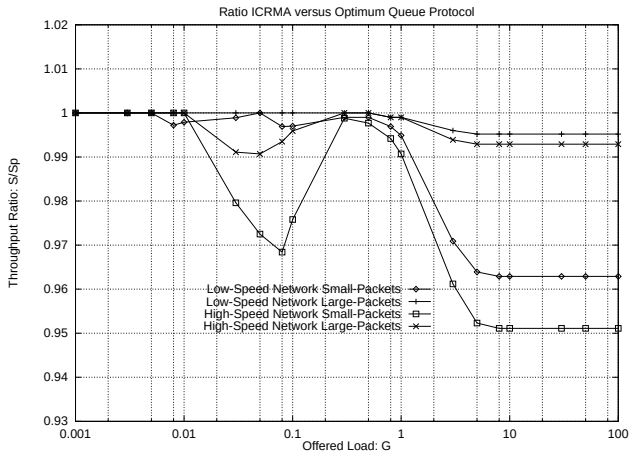

Fig. 5. Throughput achieved in ICRMA with that achieved with an ideal channel access protocol based on transmission queues and a perfect collision resolution as function of the offer load G. The maximum number of group members is $h = 32$ and the average number of packets in a message is $N = 10$.

Notice that the throughput equations are the same with the exception that average number of members in the transmission queue (ρ) as well as Π_h and Π_0 are different and are determined by the Markov chain. The average number of members in the transmission queue (ρ) increases much faster in case of the optimum transmission-queue protocol.

Eq. (16) is an exact value of the throughput, not an approximation as for the case of ICRMA, because we have used exact values for P_a and the duration of the busy period. Fig. 5 shows the ratio of the lower bound of ICRMA's average throughput given by Eq. (14) to the average throughput obtained by the optimum queue protocol given by Eq. (16) as the offered load G varies. It is evident that ICRMA approaches optimum average performance. ICRMA differs the most from the optimum MAC protocol when the average transmission queue size is small and requests for addition to the queue (RTSs) accumulate in contention periods. For the case of a high-speed network with small packets, this occurs when G is in the neighborhood of 0.1, and even in this case ICRMA exhibits an average throughput within 5% of the optimum. When data packets are much larger than RTS and CTS, ICRMA is within 1% of the optimum.

VI. Conclusions

We have described and analyzed ICRMA, a new stable multiple access protocol for broadcast channels shared by bursty stations. ICRMA dynamically divides the channel into cycles of variable length; each cycle consists of a contention period and a transmission-queue period. During the contention period, a station with one or more packets to send competes for the right to be added to the transmission group; this is done using a collision-resolution-splitting algorithm based on a request-to-send/clear-to-send (RTS/CTS) exchange with carrier sensing.

Our analysis was limited to fully connected networks using a single channel, and we have also suggested an approach to apply ICRMA to WLANs with hidden terminals. Our analysis shows that ICRMA provides high throughput when either a small or a large number of stations need to access the channel. Allowing the

maximum size of the data transmission queue to equal the number of stations in the system, ICRMA becomes TDMA in effect when all stations need to transmit. ICRMA is much more efficient than DQRAP, which is a representative of prior efficient protocols based on collision resolution. Furthermore, we have shown that ICRMA exhibits near-optimum performance. In addition to its efficiency, ICRMA is attractive, because it can operate with no need for time slotting as prior MAC protocols based on collision resolution do.

APPENDIX

Proof of Theorem 1: The initial values for $\overline{\mathcal{Z}}$ and $\overline{\mathcal{C}}$ up to $n = 4$ are given in Table I, each satisfying Eq. (1). Regardless of the value of n, $\overline{\mathcal{Z}}(n,0) = 1$, $\overline{\mathcal{Z}}(n,1) = 0$ and $\overline{\mathcal{Z}}(n,n) = 0$.

$\mathcal{Z}(1,0) = 1$	$\mathcal{Z}(3,2) = \frac{1}{3} \leq 0.886$	$\mathcal{C}(1,0) = 0$	$\mathcal{C}(3,2) = \frac{4}{3} \leq 1.886$
$\mathcal{Z}(1,1) = 0$	$\mathcal{Z}(3,3) = 0 \leq 1.299$	$\mathcal{C}(1,1) = 0$	$\mathcal{C}(3,3) = 2$
$\mathcal{Z}(2,0) = 1$	$\mathcal{Z}(4,0) = 1$	$\mathcal{C}(2,0) = 0$	$\mathcal{C}(4,0) = 0$
$\mathcal{Z}(2,1) = 0$	$\mathcal{Z}(4,1) = 0$	$\mathcal{C}(2,1) = 0$	$\mathcal{C}(4,1) = 0$
$\mathcal{Z}(2,2) = 0 \leq 0.886$	$\mathcal{Z}(4,2) = \frac{1}{3} \leq 0.886$	$\mathcal{C}(2,2) = 1 \leq 1.886$	$\mathcal{C}(4,2) = \frac{4}{3} \leq 1.886$
$\mathcal{Z}(3,0) = 1$	$\mathcal{Z}(4,3) = 0 \leq 1.299$	$\mathcal{C}(3,0) = 0$	$\mathcal{C}(4,3) = 2 \leq 3.329$
$\mathcal{Z}(3,1) = 0$	$\mathcal{Z}(4,4) = 0 \leq 1.732$	$\mathcal{C}(3,1) = 0$	$\mathcal{C}(4,4) = 3 \leq 4.772$

TABLE I

INITIAL CONDITIONS FOR THE AVERAGE NUMBER OF STEPS

Now we assume that, for all $4 \leq n \leq \alpha$ and all $2 \leq m \leq \nu$, the conditions $\overline{\mathcal{Z}}(\alpha,m) \leq 0.443m$ are satisfied, and we show that the condition holds for all $\overline{\mathcal{Z}}(n,m)$. There are three cases to consider in Eq. (1) based on the summation indices.

Case 1: $m \leq \alpha$ and $m \leq \beta$: Then $\mu = 0$ while $\nu = m$. Therefore,

$$\frac{\overline{\mathcal{Z}}(n,m)}{m} = \sum_{i=0}^{m} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m-i) + \mathcal{Z}(\beta, i)\right] \tag{17}$$

Extracting the first two and the last two elements from the summation (i.e., the elements with $i = 0, 1, m-1, m$) and noting that $\overline{\mathcal{Z}}(\alpha,1) = \overline{\mathcal{Z}}(\beta,1) = 0$, and $\overline{\mathcal{Z}}(\alpha,0) = \overline{\mathcal{Z}}(\beta,0) = 1$, we obtain

$$\begin{aligned}\overline{\mathcal{Z}}(n,m) \quad = \quad & \sum_{i=2}^{m-2} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m-i) + \mathcal{Z}(\beta, i)\right] + \\ & \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m) + 1\right] + \frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{\binom{n}{m}} \mathcal{Z}(\alpha, m-1) + \\ & \frac{\binom{\alpha}{1}\binom{\beta}{m-1}}{\binom{n}{m}} \mathcal{Z}(\beta, m-1) + \frac{\binom{\alpha}{0}\binom{\beta}{m}}{\binom{n}{m}} \left[1 + \mathcal{Z}(\beta, m)\right]\end{aligned} \tag{18}$$

Because $\overline{\mathcal{Z}}(\alpha,m) \leq 0.443m$ and $\overline{\mathcal{Z}}(\beta,m) \leq 0.443m$, we obtain

$$\begin{aligned}\overline{\mathcal{Z}}(n,m) \quad \leq \quad & 0.443m \sum_{i=0}^{m} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} - \\ & 0.443 \frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{\binom{n}{m}} - 0.443 \frac{\binom{\alpha}{1}\binom{\beta}{m-1}}{\binom{n}{m}} + \frac{\binom{\alpha}{0}\binom{\beta}{m}}{\binom{n}{m}}\end{aligned} \tag{19}$$

For any binomial coefficient, $\binom{n}{k} = \frac{n-k+1}{k}\binom{n}{k-1}$. Therefore, noticing that the sum from Eq. (19) equals one, we have

$$\begin{aligned}\overline{\mathcal{Z}}(n,m) \quad \leq \quad & 0.443m + \frac{\alpha - 0.443m\beta + 1 - m}{m} \frac{\binom{\alpha}{m-1}}{\binom{n}{m}} + \\ & \frac{\beta - 0.443m\alpha + 1 - m}{m} \frac{\binom{\beta}{m-1}}{\binom{n}{m}}\end{aligned} \tag{20}$$

For the equation $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ to be true, the last two terms in Eq. (20) must be zero or negative. If n is even, then $\alpha = \beta$ and

$$\overline{\mathcal{Z}}(n,m) \leq 0.443m + \frac{(2 - 0.886m)\alpha + (2 - 2m)}{m} \frac{\binom{\alpha}{m-1}}{\binom{n}{m}} \tag{21}$$

Because $m > 3$, $\alpha(2 - 0.886m) + (2 - 2m) < 0$; therefore, $\overline{\mathcal{Z}}(n,m) \leq 0.443m$. If n is odd, then $\beta = \alpha - 1$; accordingly, we have

$$\begin{aligned}\overline{\mathcal{Z}}(n,m) \quad \leq \quad & 0.443m + \frac{\alpha(1 - 0.443m) + 1 - 0.557m}{m} \frac{\binom{\alpha}{m-1}}{\binom{n}{m}} + \\ & \frac{\alpha(1 - 0.443m) - m}{m} \frac{\binom{\beta}{m-1}}{\binom{n}{m}}\end{aligned} \tag{22}$$

Because $m > 3$, $\alpha(1-0.443m)+1-0.557m \leq 0$ and $\alpha(1-0.443m)-m \leq 0$; therefore, our assumption that $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ is correct for any n and any $m > 1$.

Case 2: $m \leq \alpha$ and $m > \beta$: Then $\mu = 0$ while $\nu = \beta$ and n can only be odd. Further more, $\alpha = m$ and $\beta = m - 1$, while $n = 2m - 1$. Extracting the first term in the summation in Eq. (1) we obtain

$$\overline{\mathcal{Z}}(n,m) = \sum_{i=1}^{\beta} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m-i) + \mathcal{Z}(\beta, i)\right] + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} \tag{23}$$

where several terms have been eliminated since they equal zero. Because $\mathcal{Z}(m, m-i) \leq 0.443(m-i)$ and $\mathcal{Z}(m-1, i) \leq 0.443i$, we obtain

$$\overline{\mathcal{Z}}(n,m) \leq \sum_{i=1}^{\beta} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[0.443(m-i) + 0.443i\right] + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} = 0.443m \tag{24}$$

We conclude that our assumption $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ is correct for any n and all $m > 1$.

Case 3: $m > \alpha$ and $m > \beta$: Then $\mu = m - \alpha$ while $\nu = \beta$. Once again, we extract and evaluate the last member of the summation in Eq. (1) as well as set $\mathcal{Z}(\alpha, m-i) \leq 0.443(m-i)$ and $\mathcal{Z}(\beta, i) \leq 0.443i$. Therefore, we obtain

$$\overline{\mathcal{Z}}(n,m) = 0.443m - 0.443\beta \frac{\binom{\alpha}{m-\beta}\binom{\beta}{m}}{\binom{n}{m}} \leq 0.443m \tag{25}$$

We conclude that our assumption $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ is correct for any n and all $m > 1$. Therefore, for all $m > 2$ and any value of n, $\overline{\mathcal{Z}}(n,m) \leq 0.443m$. □

Proof of Theorem 2: According to Eq. (1), the $\overline{\mathcal{C}}(n,m)$ can be expressed as

$$\overline{\mathcal{C}}(n,m) = \sum_{i=\mu}^{\nu} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{C}(\alpha, m-i) + \mathcal{C}(\beta, i) + 1\right] \tag{26}$$

Case 1: $m \leq \alpha$ and $m \leq \beta$: Then $\mu = 0$ while $\nu = m$. Let us separate the first two and the last two terms from the summation and evaluate them, our expression becomes

$$\begin{aligned}\overline{\mathcal{C}}(n,m) \quad = \quad & \sum_{i=2}^{m-2} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{C}(\alpha, m-i) + \mathcal{C}(\beta, i) + 1\right] + \\ & \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} \left[\mathcal{C}(\alpha, m) + 1\right] + \frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{\binom{n}{m}} \left[\mathcal{C}(\alpha, m-1) + 1\right] + \\ & \frac{\binom{\alpha}{1}\binom{\beta}{m-1}}{\binom{n}{m}} \left[\mathcal{C}(\beta, m-1) + 1\right] + \frac{\binom{\alpha}{0}\binom{\beta}{m}}{\binom{n}{m}} \left[\mathcal{C}(\beta, m) + 1\right]\end{aligned} \tag{27}$$

Following the procedure introduced in the proof of Theorem 1, we can substitute $\mathcal{C}(\beta,i)$ and $\mathcal{C}(\alpha,i)$ by $1.443i-1$. We then proceed to collect the missing terms from the summation and arrive at the following expression:

$$\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{\binom{\alpha}{m}}{\binom{n}{m}}-1.443\beta\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}-1.443\alpha\frac{\binom{\beta}{m-1}}{\binom{n}{m}}+\frac{\binom{\beta}{m}}{\binom{n}{m}} \tag{28}$$

Which can be simplified using the binomial coefficient identity introduced in the proof of Theorem 1 to

$$\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{\alpha-1.443m\beta+1-m}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}+\frac{\beta-1.443m\alpha+1-m}{m}\frac{\binom{\beta}{m-1}}{\binom{n}{m}} \tag{29}$$

Assume that n is even, then $\alpha=\beta=\frac{n}{2}$ and

$$\overline{\mathcal{C}}(n,m) \leq \frac{3}{2}-1+2\frac{\alpha(1-1.443m)+(1-m)}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}} \leq 1.443m-1 \tag{30}$$

On the other hand, if n is odd, then $\beta=\alpha-1$, and

$$\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{\beta(1-1.443m)+(2-m)}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}+\frac{\alpha(1-1.443m)-m}{m}\frac{\binom{\beta}{m-1}}{\binom{n}{m}} \leq 1.443m-1 \tag{31}$$

Therefore, $\overline{\mathcal{C}}(n,m) \leq 1.443m-1$ for any n and all $m>1$.

Case 2: $m \leq \alpha$ and $m>\beta$: Then $\mu=0$ while $\nu=\beta$ and n can only be odd. Further more, $\alpha=m$ and $\beta=m-1$, while $n=2m-1$. Substituting all this in Eq. (1), we obtain

$$\begin{aligned}\overline{\mathcal{C}}(n,m) = &\sum_{i=2}^{\beta-1}\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{m\binom{n}{m}}[\mathcal{C}(\alpha,m-i)+\mathcal{C}(\beta,i)+1]+\\ &\frac{\binom{\alpha}{m}\binom{\beta}{0}}{m\binom{n}{m}}[\mathcal{C}(\alpha,m)+1]+\frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{m\binom{n}{m}}[\mathcal{C}(\alpha,m-1)+1]+\\ &\frac{\binom{\alpha}{m-\beta}\binom{\beta}{\beta}}{m\binom{n}{m}}[\mathcal{C}(\alpha,m-\beta)+\beta]\end{aligned} \tag{32}$$

From our induction assumption, for all α and $\beta<n$, $\mathcal{C}(\alpha,i) \leq 1.443i-1$, $\mathcal{C}(\beta,i) \leq 1.443i-1$, and from the binomial coefficient property introduced in the proof of Theorem 1, we can simplify the above equation to obtain

$$\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{\alpha-0.443\beta+1-m}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}-1.443\beta\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}} \tag{33}$$

Because $m \leq \alpha$ and $m>\beta$, n can only be odd and $\alpha=m$, while $\beta=\alpha-1$. Accordingly, we have

$$\begin{aligned}\overline{\mathcal{C}}(n,m) &\leq 1.443m-1+\frac{1.443-0.443m}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}+(1-1.443m)\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}}\\ &\leq 1.443m-1\end{aligned} \tag{34}$$

We conclude that, for any n and all $m>1$, $\overline{\mathcal{C}}(n,m) \leq 1.443m-1$ is correct.

Case 3: $m>\alpha$ and $m>\beta$: Then $\mu=m-\alpha$ while $\nu=\beta$. Therefore, Eq. (1) can be written as,

$$\overline{\mathcal{C}}(n,m)=\sum_{i=m-\alpha}^{\beta}\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}[\mathcal{C}(\alpha,m-i)+\mathcal{C}(\beta,i)+1] \tag{35}$$

Notice that terms with $i=0$ do not appear in the equation. The smallest value for i is 1. Because $\mathcal{C}(\beta,1)=\mathcal{C}(\alpha,1)=0$, it is true that $\mathcal{C}(\beta,1)=\mathcal{C}(\alpha,1) \leq 1.443m-1$; therefore,

$$\begin{aligned}\overline{\mathcal{C}}(n,m) &\leq \sum_{i=m-\alpha}^{\beta-1}\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}[1.443(m-i)-1+1.443i-1+1]+\\ &\quad\frac{\binom{\alpha}{m-\beta}\binom{\beta}{\beta}}{\binom{n}{m}}[1.443m-1.443\beta-1+\beta]\\ &= 1.433m-1-0.443\beta\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}} \leq 1.443m-1\end{aligned} \tag{36}$$

We conclude that $\overline{\mathcal{C}}(n,m) \leq 1.443m-1$ is correct for any n and all $m>1$ for all the cases. □

References

[1] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: a media access protocol for wireless LANs," *Proc. ACM SIGCOMM '94*, pp. 212–25, ACM, 1994.

[2] K. Biba, "A Hybrid Wireless MAC Protocol Supporting Asynchronous and Synchronous MSDU Delivery Services," *Tech. Rep. Paper 802.11/91-92*, IEEE 802.11 Working Group, 1992.

[3] J. Boudenant, B. Feydel and P. Rolin, "Lynx: An IEEE802.3 compatible deterministic protocol," *Proc. IEEE INFOCOM '87*, 1987.

[4] R. L. Brewster and A. M. Glass, "Throughput analysis of non-persistent and slotted non-persistent CSMA/CA protocols," *4th International Conference on Land Mobile Radio*, pp. 231–6, Institution of Electronic and Radio Engineers, 1987.

[5] J.I. Capetanakis, "Tree algorithm for packet broadcasting channel," *IEEE Trans. Inform. Theory*, vol.IT-25, pp. 505-515, Sept. 1979.

[6] C. Fullmer and J.J. Garcia-Luna-Aceves, "Solutions to Hidden-Terminal Problems in Wireless Networks," *Proc. ACM SIGCOMM '97*, Cannes, France, 1997.

[7] R. G. Gallager, "Conflict resolution in random access broadcast networks," *Proc. AFOSR Workshop Commun. Theory Appl.*, Provincetown, MA. Sept. 1978.

[8] R. Garces and J. J. Garcia-Luna-Aceves, "Floor acquisition multiple access with collision resolution," *Proc. ACM/IEEE Mobile Computing and Networking '96*, Rye, NY, Nov. 10-12, 1996.

[9] P. Karn, "MACA - a new channel access method for packet radio," *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pp. 134–40, ARRL, 1990.

[10] M. J . Karol and I. Chih-Lin, "A protocol for fast resource assignment in wireless PCS," *IEEE Transactions on Vehicular Technology*, vol. 43, no. 3, pp. 727-32, IEEE, 1994.

[11] L. Kleinrock and F. A. Tobagi, "Packet switching in radio channels: Part I - carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Trans. Commun.*, vol. COM-23, no. 12, pp. 1400–1416, 1975.

[12] J. L. Massey, "Collision resolution algorithm and random access communications," in *Multiuser Communication Systems, G. Longo Ed.*. New York: Spriger-Verlag, 1981. pp. 73-137.

[13] T. Towsley, and P. O. Vales, "Announced arrival random access protocols," *IEEE Trans. Commun.*, vol. COM-35, no. 5, pp. 513-521, May 1987.

[14] B.S. Tsybakov, and N.B. Likhanov, "Upper bound on the capacity of random multiple access system," *Problems of Information Transmission*, vol. 23, no. 3, pp. 224-236, July-Sept. 1987.

[15] W. Xu and G. Campbell, "A distributed queuing random access protocol for a broadcast channel," *Proc. ACM SIGCOMM '93*, San Francisco, CA, 13–17 Sept. 1993.